CLAIMS

1. A method for simulating a system which comprises a software element, and first and second hardware components, the software element being for execution on the second hardware component, and the first and second hardware components being operable to interact with one another, the method comprising the steps of:

simulating operation of the first hardware component in a first simulation; and

simulating the software element and the second hardware component in a second simulation;

wherein the first simulation and the second simulation are implemented in separate processing threads.

2. A method as claimed in claim 1 wherein the first simulation and the second simulation run asynchronously.

3. A method as claimed in claim 1, wherein the first simulation and the second simulation are synchronised with a reference clock.

4. A method as claimed in claim 1, wherein the first and second simulations run in different respective simulation environments.

5. A method as claimed in claim 1, further comprising:

performing operations in the first simulation to set up an inter-process communications protocol connection therein;

connecting the second simulation to the inter-process communications protocol connection in the first simulation;

connecting a software debugger to the second simulation; and

controlling the first simulation from the software debugger via the second simulation using the inter-process communications protocol.

6. A method as claimed in claim 1, further comprising:

performing operations in the first simulation to set up an inter-process communications protocol connection therein;

connecting a software debugger to the communications protocol connection; and

controlling the first simulation from the software debugger using the inter-process communications protocol.

7. A method as claimed in claim 5 or 6, wherein the inter-process communications protocol is TCP/IP and the connection is a TCP/IP socket.

8. A method as claimed claim 1, wherein the second hardware component includes a processor.

9. A method as claimed in claim 8, wherein the processor is an embedded processor.

10. A method as claimed in claim 1, wherein the hardware component includes processor peripheral devices.

11. A method as claimed in claim 10, wherein the peripheral devices are embedded.

12. A method as claimed in claim 1, wherein the first simulation is implemented using a hardware description language (HDL) simulation environment.

13. A method as claimed in claim 1, wherein the second simulation is implemented using a C model.

14. A method as claimed in claim 1, wherein the first hardware component is a programmable logic device.

15. A method for controlling a simulation of a system using a software debugger, wherein the system comprises a software element, and first and second hardware components; the software element being for execution on the second hardware component and the first and second hardware components being operable to interact with one another, the method comprising the steps of:

simulating the first hardware component in a first simulation; and

simulating the second hardware component in a second simulation;

performing operations in the first simulation to set up an inter-process communications protocol connection; and

controlling the first simulation from the software debugger using the inter-process communications protocol.

16. A method as claimed in claim 15, further comprising the step of:

connecting the software debugger to inter-process communications protocol connection.

17. A method as claimed in claim 15, further comprising the steps of:

connecting the second simulation to inter-process communications protocol connection; and

connecting the software debugger to the second simulation;

wherein the first simulation is controlled from the software debugger via the second simulation using the inter-process communications protocol.

18. A method as claimed in claim 15, wherein the inter-process communications protocol is TCP/IP and the connection is a TCP/IP socket.

19. A method as claimed in claim 15, wherein the step of simulating the second hardware component comprises simulating a processor and one or more peripheral devices with which the one or more processors interact directly.

20. A method as claimed in claim 15, wherein the first simulation and the second simulation are implemented in separate processing threads.

21. A method as claimed in claim 15, wherein the first simulation and the second simulation run asynchronously.

22. A method as claimed in claim 15, wherein the first simulation and the second simulation are synchronised with a reference clock.

23. A method as claimed in claim 15 wherein the first and second simulations are implemented in respective different simulation environments.

24. A method as claimed in claim 15, wherein the second hardware component includes embedded processors.

25. A method as claimed in claim 15, wherein the second hardware component includes embedded peripheral devices.

26. A method as claimed in claim 15, wherein the first simulation is implemented using a hardware description language (HDL) simulation environment.

27. A method as claimed in claim 15, wherein the second simulation is implemented using a C model.

28. A method as claimed in claim 15, wherein the first hardware component is a programmable logic device.

29. A method for providing an I/O interface for a simulation model to allow the simulation of interactive programs, the method comprising:

simulating an embedded input/output device within the simulation model to produce an input/output device model;

connecting the input/output device model to a terminal emulator using an inter-process communications protocol; and

running an interactive program in the terminal emulator.

30. A method as claimed in claim 29, the method further comprising:

providing separate processing threads for the embedded input/output device to allow concurrent user inputs and outputs.

31. A method as claimed in claim 29, wherein the inter-process communications protocol is TCP/IP.

32. A method as claimed in claim 29, wherein the input/output device is a UART device.

33. A method as claimed in claim 29, wherein the input/output device is an Ethernet MAC device.